What I Did Last Summer
LINGOs, GPUs, and Monitoring Vertex

Imran Haque
Department of Computer Science
Pande Lab, Stanford University

http://cs.stanford.edu/people/ihaque
http://folding.stanford.edu
ihaque@cs.stanford.edu

CUP XI
10 March 2010
A Dead White Guy

Won't you give me three steps,
Gimme three steps mister,
Gimme three steps towards the door?
Gimme three steps
Gimme three steps mister,
And you'll never see me no more.

“Gimme Three Steps”

Ronnie Van Zant,
Lead singer, Lynyrd Skynyrd
It Began in Santa Fe...

LINGO’s kind of slow... think you could put it on a GPU?
...and continued in Cambridge
### A Survey of GPU-enabled Comp Chem

<table>
<thead>
<tr>
<th>Package Name</th>
<th>Application</th>
<th>Authors/First Release</th>
<th>Speedup</th>
</tr>
</thead>
<tbody>
<tr>
<td>VMD</td>
<td>Ion placement</td>
<td>Stone et al., 2007</td>
<td>100x</td>
</tr>
<tr>
<td>TeraChem</td>
<td>Two-electron integral (quantum chem)</td>
<td>Ufimtsev and Martinez, 2008</td>
<td>130x</td>
</tr>
<tr>
<td>Folding@home/ OpenMM</td>
<td>Molecular dynamics</td>
<td>Friedrichs et al., 2009</td>
<td>735x</td>
</tr>
<tr>
<td>PAPER</td>
<td>3-D chemical similarity</td>
<td>Haque and Pande, 2009</td>
<td>30x</td>
</tr>
<tr>
<td></td>
<td>Poisson-Boltzmann solvation</td>
<td>Narumi et al., 2009</td>
<td>40x</td>
</tr>
<tr>
<td>VMD</td>
<td>MO Display</td>
<td>Stone et al., 2009</td>
<td>125x</td>
</tr>
<tr>
<td>GPU ROCS</td>
<td>3-D chemical similarity</td>
<td>OpenEye, 2010</td>
<td>120x</td>
</tr>
<tr>
<td>SIML</td>
<td>1-D chemical similarity</td>
<td>Haque, Pande, and Walters, 2010</td>
<td>83x</td>
</tr>
</tbody>
</table>
Introduction to LINGO

• “1-D” similarity method comparing canonical SMILES strings of molecules by fragmentation:
  - Benzene -> c1cccccc1 -> [c1cc, ccc1, 1ccc, cccc, cccc]
  - Pyridine -> n1cccccc1 -> [n1cc, ccc1, 1ccc, cccc, cccc]

\[
T_{A,B} = \frac{|A \cap B|}{|A \cup B|}
\]

\[
T_{A,B} = \sum_{i=1}^{\ell} \left(1 - \frac{|N_{A,i} - N_{B,i}|}{N_{A,i} + N_{B,i}}\right)
\]

• Efficient implementation by Grant et al. (2006): build DFA from reference string (O(N)), run query strings through automaton to calculate Tanimoto (also O(N))
DFAs, Big Yellow Vans, and GPUs

Coleman, *Introducing Speech and Language Processing*
DFAs, Big Yellow Vans, and GPUs
DFAs, Big Yellow Vans, and GPUs
DFAs, Big Yellow Vans, and GPUs
How to get me to work on LINGOs

Full time: Monitors: 2
Intern: Monitors: 6

WIN
A New LINGO algorithm

• Pack LINGO substrings into 32-bit integers
• Molecules $\rightarrow$ sorted lists of integers (and counts)

\[
T_{AB} = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|}
\]

• Calculate intersection by algorithm similar to merging sorted lists: simple control logic, cache friendly

Haque IH, Pande VS, Walters WP. *JCIM 2010*
CPU Performance

LINGO benchmark, 4K Maybridge, Phenom II X4
920 (2.8GHz)

SIML-CPU 2.75x the speed of OELingoSim...

*but this is no good for GPUs*
Mo’ monitors, mo’ faster
# GPU Memory Optimizations

<table>
<thead>
<tr>
<th>Mol 1</th>
<th>@H][</th>
<th>C(C)</th>
<th>C=C</th>
<th>NH0+</th>
<th>[NH0</th>
<th>COc0</th>
<th>ccc0</th>
<th>O CC=</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mol 2</td>
<td>)(C(</td>
<td>@H][</td>
<td>ccc(</td>
<td>cc0)</td>
<td>(=O)</td>
<td>O-)</td>
<td>])(O-</td>
<td>(cc0</td>
</tr>
<tr>
<td>Mol 3</td>
<td>COc(</td>
<td>@H][</td>
<td>ccc(</td>
<td>cc0)</td>
<td>(=O)</td>
<td>])(c0</td>
<td>(cc0</td>
<td>ccc0</td>
</tr>
<tr>
<td>Mol 4</td>
<td>NNC(</td>
<td>(=O)</td>
<td>O)c0</td>
<td>ccc0</td>
<td>CC(=</td>
<td>NC(=</td>
<td>]NNC</td>
<td>=O)N</td>
</tr>
</tbody>
</table>

Row-major layout is fine for the (non-vectorized) CPU because we can rely on cache to bring in partial rows for each core...

... but *kills* GPU performance

Haque IH, Pande VS, Walters WP. *JCIM* 2010
## GPU Memory Optimizations

Transposing molecule layout to column-major maximizes spatial locality among threads.

Can barrier on each row to guarantee coalescing, or use a 2D texture (if available on hardware) for more speed.

<table>
<thead>
<tr>
<th>Mol 1</th>
<th>Mol 2</th>
<th>Mol 3</th>
<th>Mol 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>@H][</td>
<td>)(C(</td>
<td>COC(</td>
<td>NNC(</td>
</tr>
<tr>
<td>C(C)</td>
<td>@H](</td>
<td>@H](</td>
<td>(=O)</td>
</tr>
<tr>
<td>C=C)</td>
<td>ccc(</td>
<td>ccc(</td>
<td>O)c0</td>
</tr>
<tr>
<td>NH0+</td>
<td>cc0)</td>
<td>cc0)</td>
<td>ccn0</td>
</tr>
<tr>
<td>[NH0</td>
<td>(=O)</td>
<td>(=O)</td>
<td>CC(=</td>
</tr>
<tr>
<td>COc0</td>
<td>O-)</td>
<td>)(c0</td>
<td>NC(=</td>
</tr>
<tr>
<td>ccc0</td>
<td>)[O-</td>
<td>(cc0</td>
<td>)NNC</td>
</tr>
<tr>
<td>0CC=</td>
<td>(cc0</td>
<td>ccc0</td>
<td>=O)N</td>
</tr>
</tbody>
</table>

Haque IH, Pande VS, Walters WP. *JCIM* 2010
GPU Performance

LINGO benchmark, 4K/32K Maybridge, Phenom II X4 920 vs Tesla T10

SIML-CUDA 83x the speed of OELingoSim...more like it!
(alpha version of SIML-OpenCL is only about 22x speedup)
Can I Have a DVD Now Please, Ant?
Who needs something this fast?

• 1K-10K molecules passes for a “large” data set in lit
• **Actually** large chemical databases:
  
  PubChem: 31M
  ZINC: 34M
  GDB-13: 970M

• Assay data exists for hundreds of thousands of mols

• I’m interested in doing **very** large-scale data integration
Example Application: PubChem

- All-vs-all, ~19.5M compounds, OE Isomeric SMILES
  380 x $10^{12}$ Tanimotos = 0.63 nmol

- Get neighbors at 4σ to define neighbor graph
- Histogram full matrix to choose significance cutoff
- Interesting graph properties?

Starting GPU computation on Compound_02800001_02825000.ism.gz
Took 91.01 ms to initialize GPULingo object with query SMILES matrices
Starting Tanimoto computation...
  Took 826.22 ms to compile 22851 query SMILES (for next frame)
  Took 447.43 ms to sync last tile (Compound_02775001_02800000.ism.gz)
Took 4844.04 ms to compute and histogram 24949 x 24436 Tanimotos on GPU
(125856 kLINGO/sec)
Similarity Properties of PubChem

• Tanimoto distribution
  4σ-equivalent at T=0.56
  24.5 x 10^9 edges in graph
  (380 x 10^{12} possible)

• Distribution is not normal
  in the positive tail
Graph Properties of PubChem – History?

- Degree distribution

Not power-law (scale-free, preferential attachment process)
Not Poisson (random)
Connected Components of PubChem

- Not all molecules are reachable from each other
- 1\textsuperscript{st} CC: 19,113,304 molecules; 2\textsuperscript{nd} CC: 692
Graph Clustering of PubChem

- Clustered using dbclus algorithm
- Cluster sizes: 104973, 69338, 61069, 51944, ...
- Limitations of LINGO
## Acknowledgments

<table>
<thead>
<tr>
<th>Stanford</th>
<th>Collaborators</th>
</tr>
</thead>
<tbody>
<tr>
<td>Vijay Pande (PI)</td>
<td>Pat Walters</td>
</tr>
<tr>
<td>Paul Novick</td>
<td>Kim Branson</td>
</tr>
<tr>
<td>Del Lucent</td>
<td>Brian Goldman</td>
</tr>
<tr>
<td>Peter Eastman</td>
<td>Richard Dixon</td>
</tr>
<tr>
<td>Mark Friedrichs</td>
<td>John Chodera</td>
</tr>
<tr>
<td>Randy Radmer</td>
<td>Michael Houston</td>
</tr>
<tr>
<td></td>
<td>Anthony Nicholls</td>
</tr>
<tr>
<td></td>
<td>Roger Sayle</td>
</tr>
</tbody>
</table>
Conclusion

- Got a large-scale LINGO problem? Try SIML. BSD (permissive open-source) license, Python bindings:
  
  [https://simtk.org/home/siml](https://simtk.org/home/siml)

- There’s (hopefully?) interesting information about chemical space and chemical biology to be learned from large-scale cheminformatics

- Any and all monitor donations will be accepted.